63-2-2

D1-82-0221

BOEING SCIENTIFIC RESEARCH LABORATORIES

A Small Procedure for Generating Normal Random Variables

G. Marsaglia

T. A. Bray

Mathematics Research

November 1962

D1-82-0221

A SMALL PROCEDURE FOR GENERATING NORMAL RANDOM VARIABLES

by

G. Marsaglia and T. A. Bray

Mathematical Note No. 283

Mathematics Research Laboratory

BOEING SCIENTIFIC RESEARCH LABORATORIES

November 1962

0. Summary

A normal random variable X may be generated in terms of uniform [0,1] random variables $u_1, u_2, \ldots$ in the following simple way: 86% of the time, put $X = 2(u_1+u_2+u_3-1.5)$, 11% of the time, put $X = 1.5(u_1+u_2-1)$, and the remaining 3% of the time, use a more complicated procedure so that the resulting mixture is correct. This method takes only half again as long as the very fastest methods, is much simpler, and requires very little storage space. It is the Volkswagen for those who can't afford a Rolls-Royce.

1. The Method

If you want to generate a normal random variable in a computer and can afford several hundred storage locations, then you should fashion a procedure based on one of the super normal methods, such as the one described in [1]. If, however, you are diffident about using a super program, then you may wish to consider an economy model. We will consider the problem of producing a normal variable in which ease of programming and small storage capacity are the primary, and running time the secondary, considerations. It turns out that our solution is almost as fast as the fastest programs.

We want a standard normal random variable X in terms of independent uniform [0,1] random variables $u, u_1, u_2, \ldots$. We seek a simple, fast method for generating X most of the time, occasionally using a more complicated correcting procedure. Our choice of the easy method is $X = 2(u_1+u_2+u_3-1.5)$. This random variable is already close to normal, and we can expect that our correcting procedure will have a comfortably small frequency. It turns out that the greatest frequency with which we can represent X in this way is $.8638$, and that, in addition, we can carve out a large portion of the residual density by putting $X = 1.5(u_1+u_2-1)$ with frequency $.1107$. More specifically, let $g_1(x)$ and $g_2(x)$ be the densities of $2(u_1+u_2+u_3-1.5)$ and $1.5(u_1+u_2-1)$, respectively:

$$g_1(x) = \begin{cases} .125(3-x^2) & |x|< 1 \\ .0625(3-|x|)^2 & 1<|x|<3 \\ 0 & 3<|x| \end{cases}$$

$$g_2(x) = \begin{cases} (1.5-|x|)/2.25 & |x|< 1.5 \\ 0 & 1.5<|x| \end{cases}$$

Then if $f(x) = e^{-\frac{1}{2}x^2}/\sqrt{(2\pi)}$, we have

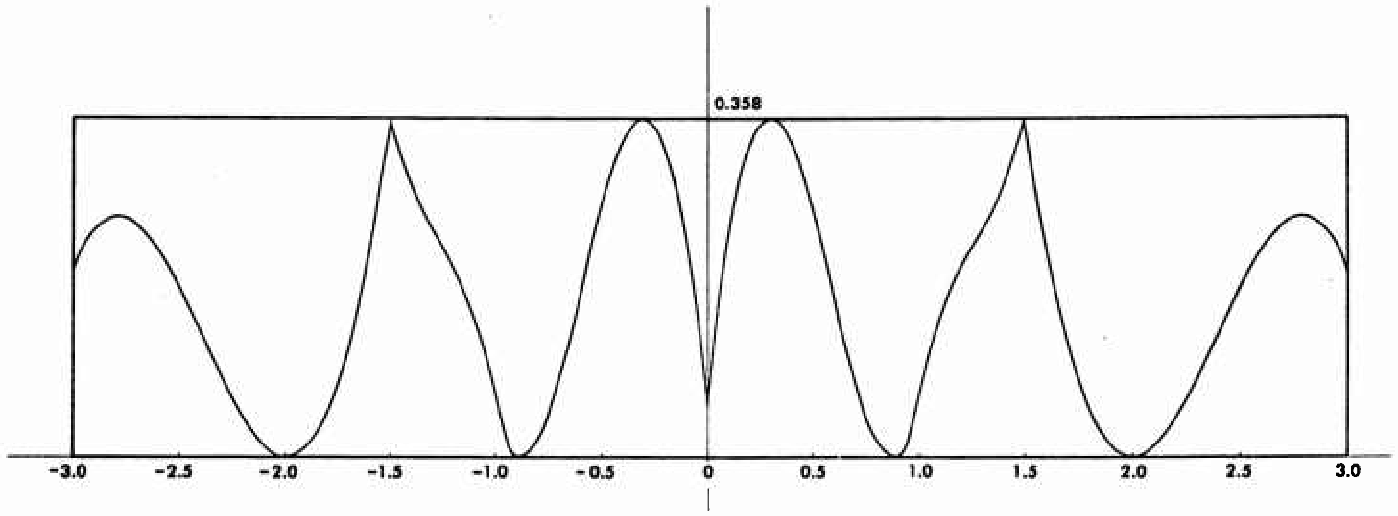
$$f(x) = .8638g_1(x)+.1107g_2(x)+.0228002039g_3(x)+.0026997961g_4(x), \tag{1}$$

where g_4 is the density of the normal tail and g_3 is the residual density:

$$g_4(x) = \begin{cases} f(x)/.0026997961 & 3<|x| \\ 0 & |x|<3 \end{cases}$$

$$g_3(x) = \begin{cases} 17.49731196e^{-\frac{1}{2}x^2}-4.73570326(3-x^2)-2.15787544(1.5-|x|), & |x|< 1 \\ 17.49731196e^{-\frac{1}{2}x^2}-2.36785163(3-|x|)^2-2.15787544(1.5-|x|), & 1<|x|< 1.5 \\ 17.49731196e^{-\frac{1}{2}x^2}-2.36785163(3-|x|)^2, & 1.5<|x|<3 \\ 0 & 3<|x| \end{cases}$$

The mixture (1) provides us with our method. We use polar coordinates to provide X with density g_4. See, for example, [1], [2]. Density g_3 is drawn in figure 1:



GRAPH OF $g_3(x)$

We may generate X with density g_3 by the rejection technique -- choose (x,y) uniformly from the rectangle and put X = x if $y<g_3(x)$ or choose a new (x,y) if $y>g_3(x)$. The efficiency of the procedure is 47%, that is, the probability that a point (x,y) chosen uniformly from the rectangle will lie under the curve $y = g_3(x)$ is .47..

Thus we have the following outline of a procedure for generating a standard normal random variable X in terms of independent uniform $[0,1]$ random variables $u_1, u_2, u_3, \ldots$:

1. With probability .8638 put $X = 2(u_1+u_2+u_3-1.5)$
2. With probability .1107 put $X = 1.5(u_1+u_2-1)$
3. With probability .0228002039, form pairs (x,y) according to:

   $x = 6u_1-3$

   $y = .358u_2$

   until $y<g_3(x)$, then put $X = x$.
4. With probability .0026997961, form pairs (x,y) according to:

   $x = v_1\{[9-2\ \ln(v_1^2+v_2^2)]/(v_1^2+v_2^2)\}^{\frac{1}{2}}$

   $y = v_2\{[9-2\ \ln(v_1^2+v_2^2)]/(v_1^2+v_2^2)\}^{\frac{1}{2}}$

   until either x or y is >3, then let that one be X. v_1 and v_2 are uniform $[-1,1]$, conditioned by $v_1^2+v_2^2 \leq 1$.

2. Comparisons

A program based on the above outline is easy to write -- steps 1 and 2 are the important ones for speed, and can be written in machine language, while one of the procedures such as FORTRAN can be used for steps 3 and 4. Now for a comparison of times and storage space. On the IBM 7090, the above, written as a function subprogram with all the linkages, resetting of index registers, returning X in normalized floating form, etc. takes about 80 cycles, whereas the super program [1], with its linkages, etc. requires about 50 cycles. Thus we may produce normal random variables in the 7090 at the rate of about 6000 per second with the above method, while the super program in [1] will provide them at the rate of about 10,000 per second. The instructions and constants for the short program require about 300 words of memory space in the 7090, whereas the super program requires about 1300.

In the IBM 1620 decimal machine, the super program takes about 20 milliseconds for each normal variable, the above method, about 35 milliseconds, both times being for function subprograms with all the necessary extras -- linkages, returning in normalized floating point, two-speed windshield wipers, etc. The instructions and constants for the short program require about 1000 core storage positions in the 1620, the super program, about 4000.

References

1. G. Marsaglia, M. D. MacLaren, T. A. Bray, "A super program for generating normal random variables", Boeing Scientific Research Laboratories Document D1-82-0219, August 1962.

2. G. Marsaglia, "Elementary relations between uniform and normal distributions in the plane", Boeing Scientific Research Laboratories Document D1-82-0197, August 1962.